Time of Arrival via Total-Variation Regularized Differentiation

Rick Chartrand, T-7, 7-8093, rickc@lanl.gov.

Description

The underlying idea is that the time of arrival is when the signal ceases to be essentially flat. Noise keeps the signal from being flat, but noise is fluctuation of a smaller scale than that of the signal proper. The method for identifying the time of arrival is to find where the derivative is essentially zero. The problem with doing this in a straightforward way is that noise in the signal will be magnified by numerical differentiation. Denoising either the result or the initial signal (or both) does not give satisfactory results. What does work is to regularize the differentiation process itself. We use total-variation regularization to find a function that is both regular and consistent with the original data. Once this is found, the time of arrival can be identified by thresholding.

Mathematical Principles

Total-variation seeks to minimize variation in a reconstructed function. In the case of simply denoising a function f, this is done by finding the minimizer u of the following quantity:

$$\int |\nabla u| + \frac{\lambda}{2} \int (u - f)^2.$$

(We use dimension-independent notation, but the application here is one-dimensional.) The parameter λ controls the relative importance of the two terms: the first term regularizes u, while the second keeps the result consistent with the data f. The ideal choice for λ is that which results in a u whose mean-squared difference from f equals the noise in f; this produces the most regular u that is consistent with the data.

We combine this with operator-inversion by simply incorporating the operator into the data-fidelity term:

$$\int |\nabla u| + \frac{\lambda}{2} \int (Pu - f)^2.$$

This is particularly suitable in the case that P^{-1} is noise-increasing. In our case, P will be the operator of antidifferentiation: $Pu(x) = \int_0^x u$. The DDMA team has previously implemented this in the case where P is the Abel transform.

We compute the minimizer by gradient descent. This amounts to evolving to stationarity the PDE

$$u_t = \nabla \cdot \frac{\nabla}{|\nabla|} - \lambda\big(P^* Pu - P^* f\big).$$

When P is antidifferentiation, the adjoint is $P^* u(x) = \int_x^N u$, where N is the length of the signal.

Once the regularized derivative u is found, thresholding determines where the signal is essentially constant. We seek the last such interval of appropriate length, to determine when the signal last leaves the noise floor.

Usage

Several parameters must be chosen. Some affect results, while others affect running time. This is an issue particularly due to gradient descent being fundamentally slow. More sophisticated, faster TV-minimization procedures are known, but could not be implemented in the available time.

Outer algorithm parameters are:

`N`: the length of initial signal to use. Must exceed the latest expected TOA, but smaller values make for shorter running time.

`n`: the length of initial to signal to use for estimating the noise level. Longer makes for a more robust estimate, but must not exceed the length of the noise floor. At present, the main use of the noise estimate is for hand-adjusting the regularization parameter.

`window`: the length of interval shorter than which any flat portion will be deemed spurious. Smaller values increase the risk of finding late-time flat portions in the main signal. Larger values increase the risk that blips prior to the true TOA will be regarded as heralding the main signal.

`cutoff`: the signal is truncated once it reaches this value for an interval of length `window`. This greatly speeds convergence. It must be large enough not to affect the noise floor. Changing this value affects optimal values of `ep_J` and `lam`. Truncating in this way increases the risk of over-regularizing the beginning of the signal to the point of it looking flat. Best results would come from not truncating, but this would increase the running time to a handful of minutes per signal, which was not feasible in the timeframe of this work.

`thresh`: the derivative threshold, below which the signal is considered essentially flat.

Parameters for the inner TV-minimization script are:

`ep_J`: the mean change in the computed derivative J below which the TV-minimization is regarded as sufficiently close to convergence. A too large value can give incorrect results. The smaller the value, the longer the running time.

`lam`: the regularization parameter λ. This is presently hand tuned on a subset of the prepended data. Algorithms exist for estimating λ automatically.

`iter`: the number of TV-flow timesteps to do before checking for convergence. Not very important.

`dt`: the stepsize for the PDE. Decreasing this makes for longer running time. A too-large value makes the PDE numerically unstable.

`eps`: the PDE has a singularity where $\nabla u = 0$. Adding eps to the denominator removes this, at the cost of decreasing accuracy a little.

All the parameters were hand tuned on a subset of the prepended strain gauge data. Data of different scale or noise characteristics will require parameter adjustments. It is possible to make this process automatic.

Comparing the results with twelve samples for which an "expert TOA" is available, we have the following:

```
expert - tvdertoa.m:
   17    9   15    13    -1    13    10   22   1    14   23   15
```

The computed TOA averages 12.58 samples earlier. The earliness reflects the fact that the algorithm identifies when the trend in the data will take it out of the noise band, before it actually leaves. The noise in the data can make this trend difficult to identify by eye, so it can be argued that a hand-identification of TOA from the original data gives a result slightly late. Regularizing the derivative allows for the trend to be identified despite the noise.

The files `toaresults.txt` and `toaresults.mat` contain the results on the full prepended strain gauge data. Some very late times occur when the data contains unusual flat portions after the main signal has arrived. This could be solved by truncating the signal soon after the signal has clearly arrived. The code attempts to do this using a fixed threshold on the signal values; occasionally this fails to remove a large segment of signal that includes a flat portion. Further work should be able to remove this problem. The other substantial issue is the hand tuning of the parameters to be suitable for a subset of the data. It would be possible to automatically generate these parameters for any given signal; this would increase accuracy significantly.